Bug Report 1

Executive Summary

API accepts JSON even when Content-Type is application/xml

Description of Bug

The Todo API successfully processes JSON request bodies even when the `Content-Type` header is set to `application/xml`. This contradicts the documented behavior, which states that XML content types should be used only when sending XML. The system ignores the header and parses the body as JSON.

Potential Impact

Clients may unknowingly send incorrectly formatted requests that still succeed, leading to inconsistent integrations, hidden client bugs, and reduced protocol reliability. This behavior can also create security and validation risks.

Steps to Reproduce

1. Start the server
2. Run:

```
curl -X POST http://localhost:4567/todos \
-H "Content-Type: application/xml" \
-d '{"title":"confused"}'
```

3. Observe that the todo is created successfully.

Bug Report 2

Executive Summary (≤ 80 chars)

API accepts JSON requests without any Content-Type header

Description of Bug

The API processes JSON bodies even when no `Content-Type` header is provided. The server defaults to JSON parsing without validating the request header.

Potential Impact

This weakens protocol enforcement and allows malformed or ambiguous requests to succeed, increasing the likelihood of integration errors and inconsistent client behavior.

Steps to Reproduce

1. Start the server
2. Run:

```
curl -X POST http://localhost:4567/todos \
-d '{"title":"nocontenttype"}'
```

3. Observe that the todo is created successfully.

Bug Report 3

Executive Summary (≤ 80 chars)

Java parsing exceptions are exposed in API error responses

Description of Bug

When invalid content is sent (e.g., XML body with JSON content-type), the API returns a raw Java exception message instead of a structured API error.

Potential Impact

This exposes internal implementation details, reduces professionalism of the API, complicates client error handling, and may pose security risks.

Steps to Reproduce

1. Start the server
2. Run:

```
curl -X POST http://localhost:4567/todos \
-H "Content-Type: application/json" \
-d '<todo><title>confused</title></todo>'
```

3. Observe the Java exception text in the response.

Bug Report 4

Executive Summary (≤ 80 chars)

Relationship POST endpoints reject ID despite documentation

Description of Bug

The API documentation indicates that relationships can be created using POST requests with an `id` in the body. However, the system rejects such requests with a validation error.

Potential Impact

Developers relying on documentation cannot create relationships as described. This blocks documented functionality and causes integration failures.

Steps to Reproduce

1. Start the server
2. Run:

```
curl -X POST http://localhost:4567/todos/1/categories \
-H "Content-Type: application/json" \
-d '{"id":"1"}'
```

3. Observe validation failure.

Bug Report 5

Executive Summary (≤ 80 chars)

Unsupported HTTP methods return HTML instead of API errors

Description of Bug

Unsupported methods such as PATCH and TRACE return HTML error pages rather than JSON-formatted API error responses.

Potential Impact

This breaks API consistency, complicates client error handling, and exposes framework-level responses instead of controlled API behavior.

Steps to Reproduce

1. Start the server
2. Run:

curl -X TRACE http://localhost:4567/todos

3. Observe HTML error page.

Bug Report 6

Executive Summary (≤ 80 chars)

Relationship deletion endpoints return no confirmation body

Description of Bug

DELETE requests on `/todos` relationship endpoints succeed silently and return no confirmation payload or structured response.

Potential Impact

Clients cannot easily verify whether deletion occurred, increasing the risk of inconsistent client state and difficult debugging.

Steps to Reproduce

1. Start the server
2. Run:

curl -X DELETE http://localhost:4567/todos/1/categories/1

3. Observe empty response body.